AMENDMENTS TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings, of claims in the application:

LISTING OF THE CLAIMS:

1. (Currently amended) A data processing system comprising:

a. a plurality of data processing system components, the components each responsible for carrying out a subset of data processing system functions;

b. a plurality of function domains, the domains having associated with them subset of the data processing system functions, with a plurality of peer domain level components thus carrying out the data processing functions for a given domain, and the plurality of domains forming a domain hierarchy, at least one component of at least one domain at a given level in the hierarchy providing a failure notification, which is of a hang state detected by a components peer element; and

c. a system availability manager comprising:

i. a plurality of Available Manager (AM) elements, each AM element associated with a corresponding one of the data processing system components, the AM elements thus also arranged in an AM hierarchy that parallels the domain hierarchy, at least one AM element connected to receive failure notification from one or more AM elements associated with the data processing system components associated with a next lower domain level, peer AM elements participate in a heartbeat protocol to detect a component hang state such that during their normal state execution, wherein peer AM elements located at the same level in the AM hierarchy provide a heartbeat signal notification in a determined sequence to at least one of its peer AM elements.

2. (Currently amended) A system as in claim 1 wherein ~~the~~ each AM element determines if the component in the next lower domain level from which a failure notification was received can be restarted.

3. (Currently amended) A system as in claim 2 wherein ~~the AM element~~ each of the AM elements additionally determines if the failure-notifying component can be restarted, and if it can

be restarted, ~~the AM element~~ the AM element making the determination causes that component to be restarted, without notifying a higher level AM element in the AM hierarchy.

4. (Original) A system as in claim 1 wherein the failure notification is caused by termination of processing by the monitored component.

5. (Original) A system as in claim 1 wherein the failure notification is caused by an error state in the monitored component.

6. (Currently amended) A system as in claim 3 wherein if ~~the AM element~~ any of the AM elements determines that the failure-notifying component cannot be restarted, the determining AM element sends a failure notification to a higher AM component in the AM hierarchy.

7. (Original) A system as in claim 1 wherein the AM element failure notification is constrained to the next higher AM element in the hierarchy.

8. (Original) A system as in claim 1 wherein the AM element failure notification is constrained to the next higher level AM element in the hierarchy, such that a higher level AM element in the hierarchy will then be given control over determining whether to send further failure notifications up the AM element hierarchy.

9. (Original) A system as in claim 1 wherein the data processing system components are both hardware and software components.

10. (Original) A system as in claim 9 wherein the software components include, but are not limited to, operating system hardware.

11. (Original) A system as in claim 9 wherein the software components comprise application program processes.

12. (Original) A system as in claim 1 wherein the failure notification is a component execution termination notice.

13. (Original) A system as in claim 1 wherein the failure notification is a hang state notice.

14. (Cancelled)

15. (Cancelled)

16. (Currently amended) A system as in claim ~~15~~ 1 wherein the heartbeat protocol is carried out for AM elements associated with a one of a card manager (CM) level or a system manager (SM) level in the AM hierarchy.

17. (Original) A system as in claim 16 wherein at least one of the peer AM elements reports the hang state by ceasing to send update signals to a hardware component.

18. (Original) A system as in claim 17 wherein the hardware component is a watchdog timer component.

19. (Original) A system as in claim 18 wherein the update signals are time slotted, such that a given AM element is assigned a predetermined slot in which the watchdog timer component expects to receive an update signal.

20. (Original) A system as in claim 19 wherein a failure to detect a hang state results in restarting the watchdog timer.

21. (Original) A system as in claim 19 wherein the update signal is a data word unique to the reporting AM element.

22. (Currently amended) A system as in claim ~~15~~ 1 wherein prior to the failure notification, at least one of a given group of data processing system components stores state information in a persistent storage medium.

23. (Original) A system as in claim 22 wherein the state information storage is program driven, event driven, periodically driven, or termination triggered.

24. (Original) A system as in claim 22 wherein the state information as selected from the group consisting of a machine state, application configuration state, or application data state.

25. (Original) A system as in claim 2 wherein the failure notification includes information as to whether the logical state of the component itself indicates the component can be restarted.

26. (Original) A system as in claim 2 wherein components are classified in the hierarchy with regard to a potential severity indication of their failure modalities.

27. (Currently amended) A system as in claim ~~15~~ 1 wherein the heartbeat protocol is not carried out for AM elements associated with data processing components having specific hang detection logic.

28. (Original) A system as in claim 19 wherein the heartbeat signal is a read of a unique value by the watchdog timer.

29. (Currently amended) A system as in claim 2 wherein ~~the AM element~~ each of the AM elements determines whether the associated component itself can be restarted without affecting operation of other data processing system components.

30. (Currently amended) A system as in claim 2 wherein at least some of the components of the data processing system are operating system components, ~~the AM element~~ each of the AM elements runs an application space as a process under an operating system, and the failure notification is made by signaling the associated AM element through an operating system

message.

31. (Currently amended) ~~A system as in claim 7 wherein~~ A data processing system comprising:

a. a plurality of data processing system components, the components each responsible for carrying out a subset of data processing system functions;

b. a plurality of function domains, the domains having associated with them subset of the data processing system functions, with a plurality of peer domain level components thus carrying out the data processing functions for a given domain, and the plurality of domains forming a domain hierarchy, at least one component of at least one domain at a given level in the hierarchy providing a failure notification; and

c. a system availability manager comprising:

i. a plurality of Available Manager (AM) elements, each AM element associated with a corresponding one of the data processing system components, the AM elements thus also arranged in an AM hierarchy that parallels the domain hierarchy, at least one AM element connected to receive failure notification from one or more AM elements associated with the data processing system components associated with a next lower domain level, the AM element failure notification is constrained to the next higher AM element in the hierarchy, wherein the AM failure notification to the higher level AM component in the AM hierarchy causes the AM elements that are peers of the failure-notifying AM element to be terminated.

32. (Cancelled)

33. (Currently amended) ~~A system as in claim 32 wherein~~ A data processing system comprising:

a. a plurality of data processing system components, the components each responsible for carrying out a subset of data processing system functions;

b. a plurality of function domains, the domains having associated with them subset of the data processing system functions, with a plurality of peer domain level components thus carrying out the data processing functions for a given domain, and the plurality of

domains forming a domain hierarchy, at least one component of at least one domain at a given level in the hierarchy being restartable; and

c. a system availability manager comprising: a plurality of Availability Manager (AM) elements, each AM element associated with a corresponding one of the data processing system components, the AM elements thus also arranged in an AM hierarchy that parallels the domain hierarchy, at least one AM element connected to restart one or more AM elements associated with the data processing system components associated with a next lower domain level, the at least one AM element may access operating system component state information regarding whether an operating system underlying ~~the~~ a terminating component of the plurality of data processing system components considers that the component can be restarted.

34. (Currently amended) A system as in claim 33 wherein the ~~termination notice~~ terminating component itself contains the operating system component state information.

35. (Currently amended) A system as in claim 34 wherein the at least one AM element uses the operating system component state information to determine whether the component can be restarted.

36. (Currently amended) A system as in claim ~~32~~ 33 wherein upon receiving a termination notice, the AM element may access component originated state information regarding whether according to the components own logic, the component can be restarted.

37. (Original) A system as in claim 36 wherein the termination notice itself contains the component originated state information.

38. (Original) A system as in claim 37 wherein the AM element may also access operating system component state information regarding whether an operating system underlying the terminating component considers that the component can be restarted.

39. (Original) A system as in claim 38 wherein the AM element first uses the operating system

component state information to determine whether the component can be restarted, and such indication is positive, then the AM element uses the component originated state information.

40. (Currently amended) ~~A system as in claim 32 wherein the data process system component domains~~ A data processing system comprising:

a. a plurality of data processing system components, the components each responsible for carrying out a subset of data processing system functions;

b. a plurality of function domains, which are selected from the group consisting of system, card, processor, process, and application process threads, the domains having associated with them subset of the data processing system functions, with a plurality of peer domain level components thus carrying out the data processing functions for a given domain, and the plurality of domains forming a domain hierarchy, at least one component of at least one domain at a given level in the hierarchy being restartable; and

c. a system availability manager comprising: a plurality of Availability Manager (AM) elements, each AM element associated with a corresponding one of the data processing system components, the AM elements thus also arranged in an AM hierarchy that parallels the domain hierarchy, at least one AM element connected to restart one or more AM elements associated with the data processing system components associated with a next lower domain level.

41. (Cancelled)

42. (Currently amended) A system as in claim ~~41~~ 40 wherein ~~the~~ at least one thread domain does not have associated AM elements.

43. (Cancelled)

44. (Currently amended) ~~A system as in claim 43 wherein the internal state information~~ A data processing system comprising:

a. a plurality of data processing system components which maintain internal state information, which is selected from the group consisting of processor execution state,

configuration state, and application data state, in persistent storage to permit warm restart processing, the components each responsible for carrying out a subset of data processing system functions;

b. a plurality of function domains, the domains having associated with them subset of the data processing system functions, with a plurality of peer domain level components thus carrying out the data processing functions for a given domain, and the plurality of domains forming a domain hierarchy, at least one component of at least one domain at a given level in the hierarchy being restartable; and

c. a system availability manager comprising: a plurality of Availability Manager (AM) elements, each AM element associated with a corresponding one of the data processing system components, the AM elements thus also arranged in an AM hierarchy that parallels the domain hierarchy, at least one AM element connected to restart one or more AM elements associated with the data processing system components associated with a next lower domain level.

45. (Original) A system as in claim 44 wherein the system is deployed as a networking device, and the internal state information is selected from the groups consisting of routing table, forwarding table, switching table, or other networking configuration data.

46. (Currently amended) A system as in claim ~~43~~ 44 wherein an operating system component reclaims resources upon termination of a process element.

47. (Currently amended) A system as in claim ~~43~~ 44 wherein an operating system component maintains state information regarding resources in use by executing processes.

48. (Original) A system as in claim 47 wherein the operating system derives information regarding whether a process can be restarted by examining the state information regarding resources in use.

49. (Currently amended) A system as in claim ~~43~~ 44 wherein if component failure causes inconsistency in internal operating system state, the associated processes are considered to be

nonrestartable.

50. (Currently amended) A system as in claim ~~43~~ 44 wherein a component will be subjected to a warm or cold restart process depending upon whether complete stored state information is available and valid.

51. (Currently amended) A system as in claim ~~32~~ 33 wherein the AM element hierarchy includes a system manager root level, process manager child level, and card manager parent level in the AM element hierarchy.

52. (Currently amended) ~~A system as in claim 32 wherein~~ A data processing system comprising:

a. a plurality of data processing system components, the components each responsible for carrying out a subset of data processing system functions;

b. a plurality of function domains, the domains having associated with them subset of the data processing system functions, with a plurality of peer domain level components thus carrying out the data processing functions for a given domain, and the plurality of domains forming a domain hierarchy, at least one component of at least one domain at a given level in the hierarchy being restartable; and

c. a system availability manager comprising: a plurality of Availability Manager (AM) elements, each AM element associated with a corresponding one of the data processing system components, the AM elements thus also arranged in an AM hierarchy that parallels the domain hierarchy, the AM element hierarchy includes a process manager root level, thread child level, and system manager parent level in the AM element hierarchy, and at least one AM element connected to restart one or more AM elements associated with the data processing system components associated with a next lower domain level.

53. (Currently amended) A system as in claim ~~32~~ 33 wherein the AM element hierarchy includes a card manager root level, system manager child level, and watchdog timer parent level in the AM element hierarchy.

54. (Currently amended) A data processing system comprising:

a. a plurality of data processing system components, the components comprising system cards, processors, and software processes that execute on the processors, the components thus forming a function domain hierarchy;

b. a plurality of Availability Manager (AM) elements, each AM element associated with at least one of the data processing system components, the AM elements also arranged in a hierarchy that parallels the domain hierarchy such that a card manager (CM) element in the AM hierarchy is associated with a system card component; a system manager (SM) element in the AM hierarchy is associated with a processor component; a process manager (PM) in the AM hierarchy is associated with a software process component; with at least one of the AM elements participating in an identity management protocol; with at least one of the AM elements connected to restart components associated with a next lower domain level.

55. (Cancelled)

56. (Currently amended) A system as in claim ~~55~~ 54 wherein card manager (CM) element of the AM hierarchy performs an identity management protocol for system card components.

57. (Original) A system as in claim 56 wherein the identity management protocol identifies which system card is to be considered a master.

58. (Currently amended) ~~A system as in claim 54~~ A data processing system comprising:

a. a plurality of data processing system components, the components comprising system cards, processors, and software processes that execute on the processors, the components thus forming a function domain hierarchy;

b. a plurality of Availability Manager (AM) elements, each AM element associated with at least one of the data processing system components, the AM elements also arranged in a hierarchy that parallels the domain hierarchy such that a card manager (CM) element in the AM hierarchy is associated with a system card component; a system manager (SM)

element in the AM hierarchy is associated with a processor component; a process manager (PM) in the AM hierarchy is associated with a software process component; with at least one of the AM elements connected to restart components associated with a next lower domain level, wherein if a given root element, R, in the AM hierarchy has at least one child element, C, and at least one parent element, P, the root element R is responsible for determining a failure notification only for its child elements C, and reporting the fact of a failed child AM element C to the parent element P if the child element C cannot be restarted.

59. (Original) A system as in claim 58 wherein a decision regarding whether a given child element C can be restarted is made from information provided by the child C with the failure notification.

60. (Original) A system as in claim 58 wherein a decision regarding whether a given child element C can be restarted is made from information provided by peer AM elements to the root element R.

61. (Original) A system as in claim 58 wherein a sub-group, X, of child element C are considered to have restart dependency, and upon failure notification of one element C in the group X, restarting all other elements C that are a member of the group X.

62. (Original) A system as in claim 61 wherein the elements C that are not members of the group X are not restarted.

63. (Original) A system as in claim 61 wherein the fact of the failed element C is not reported to the parent element P if all members of the group X can be restarted.

64. (Currently amended) A system as in claim 54 wherein peer AM elements detect a hung component by sending ~~the~~ a heartbeat signal at a determined time period interval.

65. (Currently amended) A system as in claim 64 wherein the ~~heartbeat time period~~ determined

time period interval at which the heartbeat signal is sent is less than a time-out duration associated with a watchdog timer that is monitoring at least one AM element.

66. (Original) A system as in claim 54 wherein a hardware component detects an event that all AM elements at the same level hang.

67. (Original) A system as in claim 64 wherein a heartbeat timeout threshold is selected so that if at least one AM element among a set of peer AM elements is hung, the peer AM element that detected the hang state will detect and record the hung state before expiration of a watchdog timer.

68. (Original) A system as in claim 64 wherein the heartbeat signals coupled from an AM element to a heartbeat register associated with the watchdog timer are time slotted.

69. (Original) A system as in claim 64 wherein the AM element and its peers are located at a system manager (SM) level in the AM hierarchy associated with monitoring processor components.

70. (Currently amended) A system as in claim ~~1~~ 54 wherein a card manager (CM) level in the AM hierarchy is associated with a system card component, and a system manager (SM) level in the AM hierarchy is associated with a processor component, and wherein a failure notification by a card manager (CM) element is provided to a watchdog timer element.

71. (Original) A system as in claim 70 wherein the watchdog timer is connected to initiate a reset for selected components of the data processing system.

72. (Currently amended) A system as in claim ~~72~~ 71 wherein a reset initiated by the watchdog timer starts an identity management process in one of the components.

73. (Original) A system as in claim 72 wherein the identity management process executes in one of the AM elements.

74. (Original) A system as in claim 72 wherein the data processing system components comprise system cards, and the identity management process is executed to identify a master system card.

75. (Original) A system as in claim 74 wherein the identity management process identifies the master system card without using a prior central system resource set.

76. (Original) A system as in claim 75 wherein the identity management process uses information concerning physical location identification of the system cards, system card present information, and a register write protocol that ensures singular assertion of master state for a given system card.

77. (Original) A system as in claim 76 wherein the identity manager process ensures singular assertion of the master state by the process of a. setting a temporary master state; b. waiting a predetermined period of time; c. setting a final master state only if no other system card has asserted temporary or master state.

78. (Currently amended) A method for determining which of a plurality of data processing system components are to assert a master state, and which are to assert a non-master state after a system reset with no retention of prior state information, the process comprising a join protocol that is executed on each system component, the join protocol comprising the steps of:

a. entering an initialization state;

b. determining a physical position for the component with respect to other components in the system;

c. determining a physical position for the component with respect to a card slot location in a backplane;

~~c.~~ d. reading state information as to master state assertions by other components;

~~d.~~ e. temporarily initializing a local master state register to the asserted state, if no other component has asserted the master state;

~~e.~~ f. waiting a predetermined pause period;

~~f.~~ g. reading other components master state assertions; and

~~g.~~ h. committing to assume the master state for further execution should no other component have asserted the master state during the waiting period.

79. (Cancelled)

80. (Currently amended) ~~A method as in claim 78~~ A method for determining which of a plurality of data processing system components are to assert a master state, and which are to assert a non-master state after a system reset with no retention of prior state information, the process comprising a join protocol that is executed on each system component, the join protocol comprising the steps of:

a. entering an initialization state;

b. determining a physical position for the component with respect to other components in the system;

c. reading state information as to master state assertions by other components;

d. temporarily initializing a local master state register to the asserted state, if no other component has asserted the master state;

e. waiting a predetermined pause period, wherein the pause period is greater than the longest expected initialization process for components in the system to read a master state register, determine a state that it should assume, and assume the determined state;

f. reading other components master state assertions; and

g. committing to assume the master state for further execution should no other component have asserted the master state during the waiting period.

81. (Original) A method as in claim 78 wherein the step of committing to assume the master state is as a Physical Default Master state.

82. (Original) A method as in claim 81 wherein the step of committing to assume the master state additionally comprises the step of: transitioning to a Logical Default Master if the read of the master state assertions by the other components indicates that no other component has asserted the master state.

83. (Currently amended) A method as in claim 81 wherein while in ~~the~~ a Logical Default Master mode, additionally comprising the steps of: in the commit step, determining if any other component has a higher priority location with a master state asserted; waiting a secondary pause period; and if so, then de-asserting the master state and committing to a non-master state.

84. (Currently amended) ~~A method as in claim 78 additionally comprising the step of:~~ A method for determining which of a plurality of data processing system components are to assert a master state, and which are to assert a non-master state after a system reset with no retention of prior state information, the process comprising a join protocol that is executed on each system component, the join protocol comprising the steps of:

a. entering an initialization state;

b. determining a physical position for the component with respect to other components in the system;

c. determining a physical position for the component with respect to respect to a card slot location in a backplane;

d. reading state information as to master state assertions by other components;

e. temporarily initializing a local master state register to the asserted state, if no other component has asserted the master state;

f. waiting a predetermined pause period;

g. reading other components master state assertions;

h. committing to assume the master state for further execution should no other component have asserted the master state during the waiting period; and

i. executing a Depart State Machine upon receipt of a reset command from another data processing system component.

85. (Currently amended) A method as in claim ~~78~~ 84 wherein the Depart State Machine comprises the step of: determining if a departed component was previously the master; and if so, executing the join ~~state machine~~ protocol.

86. (Original) A method as in claim 84 wherein the reset command is received from a watchdog timer.

87. (Original) A method as in claim 84 wherein the reset command is invoked by a restart decision made by an AM element.

88. (Original) A method as in claim 86 wherein a watchdog timer issues the reset command upon termination notification from an Availability Manager (AM) process.

89. (Original) A method as in claim 88 wherein the AM process is a distributed hierarchy of AM processes having loosely coupled AM process elements that monitor corresponding data processing system components.

90. (Original) A method as in claim 89 wherein the AM element hierarchy includes a card manager root level, system manager child level, and watchdog timer parent level in the AM element hierarchy.

91. (Original) A method as in claim 89 wherein the AM element hierarchy includes inter-card, intra-card, and processor levels.

92. (Original) A method as in claim 89 wherein the data processing system components comprise system cards, processors, and software processes associated with inter-card, intra-card, and processor levels in the AM hierarchy, respectively, and a card manager (CM) level in the AM hierarchy is associated with a system card component, a system manager (SM) level in the AM hierarchy is associated with a processor component, a system manager (SM) level in the AM hierarchy is associated with an operating system software component, and a process manager (PM) level in the AM hierarchy is associated with an executing application process software component.

93. (Original) A method as in claim 89 wherein one or more of the AM elements participate in an identity management protocol.

94. (Currently amended) A hierarchical, distributed, loosely coupled Availability Management

(AM) method for recovering from failure of execution ~~of a data processing system~~ of one or more data processing system components, wherein the data processing system components include at least two or more system cards, and at least two AM element processes are associated with monitoring the status of at least two of the system cards, the process comprising the steps of:

a. executing a plurality of AM element processes in a multi-tasking environment, the AM element processes arranged in a hierarchy, with the hierarchy of the AM elements corresponding to a failure modality hierarchy of the data processing system components;

b. within a given AM element process, receiving a termination notice from one of the data processing system components;

c. if the data processing system component can be restarted by the AM element process, then restarting the component;

d. if the data processing system component cannot be restarted by the AM element process, providing a termination notice to a higher level AM element process; ~~and~~

e. terminating execution of the AM element process;

f. if one of the system cards provides a termination indication to its associated AM process, determining if the system card can be restarted;

g. if the system card can be restarted, then asserting a system card restart command;

h. if the system card cannot be restarted by the AM process, asserting a system reset signal; and

i. thereby constraining restarts of the entire system only to instances where lower level component parts cannot be restarted.

95. (Cancelled)

96. (Currently amended) A method as in claim ~~95~~ 94 additionally comprising: failing over to a second system card if in step (h) the system card can be restarted.

97. (Currently amended) A method as in claim ~~95~~ 94 wherein an identity management protocol is executed as part of restarting a higher level component.

98. (Currently amended) A method as in claim ~~95~~ 94 wherein peer AM elements participate in a heartbeat protocol to detect a component hang state in other peered AM elements.

99. (Original) A method as in claim 98 wherein the heartbeat protocol is providing a heartbeat signal in a determined sequence to peer AM elements.